Welcome to The Logic Design Lab!

Fall 2019
Lab 1: Gate-Level Verilog

Prof. Chun-Yi Lee

Department of Computer Science
National Tsing Hua University

Agenda

- Lab 1 Outline
- Lab 1 Basic Questions
- Lab 1 Advanced Questions
- Basic Concept of Verilog Testbench

Lab 1 Outline

- Basic questions (1.5%)
  - Individual assignment
  - Due on 9/12/2019. In class.
  - Only demonstration is necessary. Nothing to submit.

- Advanced questions (5%)
  - Group assignment
  - ILMS submission due on 9/19/2019. 23:59:59.
  - Demonstration on your FPGA board (In class)
  - Assignment submission (Submit to ILMS)
    - Source codes and testbenches
    - Lab report in PDF

Lab 1 Rules

- Only gate-level description is permitted
  - Only basic logic gates are ALLOWED (AND, OR, NAND, NOR, NOT)
  - Sorry, no xor & xnor

- Please AVOID using
  - Continuous assignment and conditional operators
  - Behavioral operators (e.g., +, -, &, |, ^, &&, !, ~….., etc.)

Lab 1 Submission Requirements

- Source codes and test benches
  - Please follow the templates EXACTLY
  - We will test your codes by TAs' testbenches

- Lab 1 report
  - Please submit your report in a single PDF file
  - Please draw the gate-level circuits of your designs
  - Please explain your designs in detail
  - Please list the contributions of each team member clearly
  - Please explain how you test your design
  - What you have learned from Lab 1

Agenda

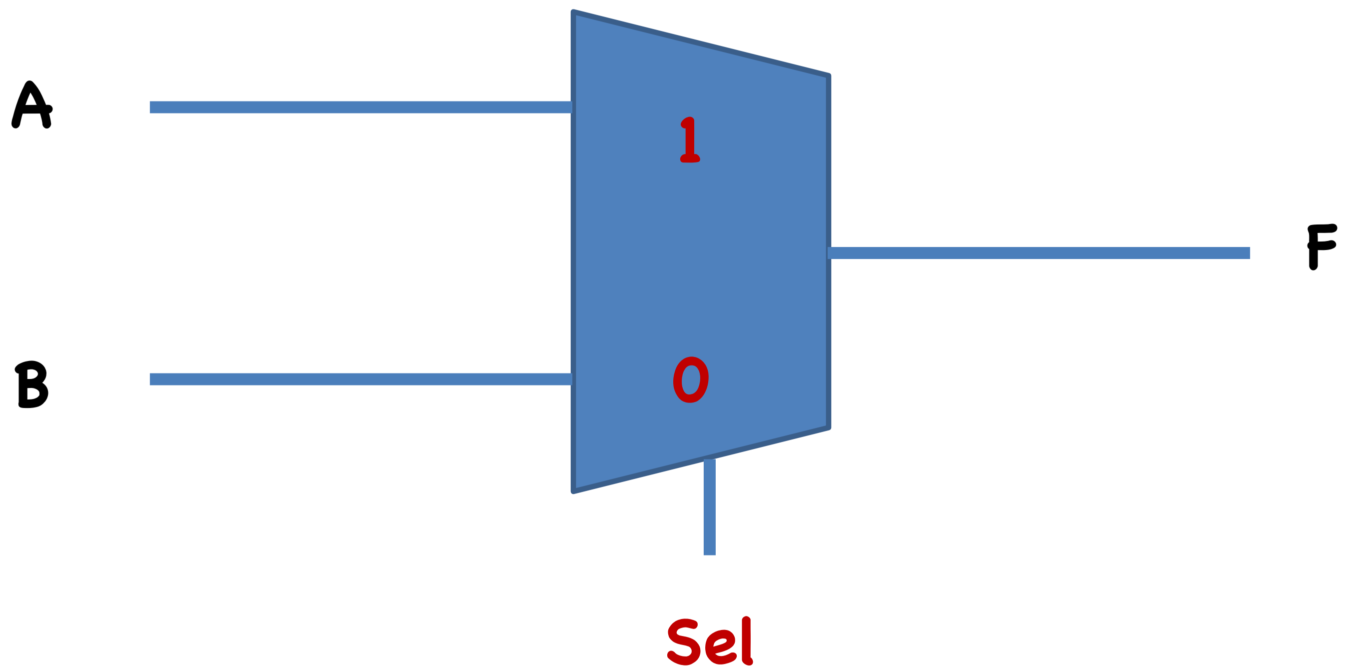
- Lab 1 Outline
- Lab 1 Basic Questions
- Lab 1 Advanced Questions
- Basic Concept of Verilog Testbench

Basic Questions

- Individual assignment
- Verilog questions (due on 9/12/2019. In class.)
  - (Gate-level) 1-bit 2-to-1 multiplexer (abbreviated as MUX)
  - (Gate-level) 1-bit full adder
- Demonstrate your work by waveforms

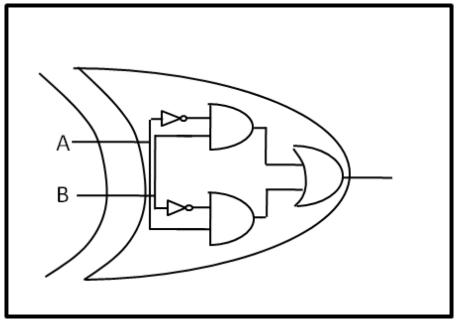
Verilog Question 1

- (Gate-level) 1-bit 2-to-1 MUX

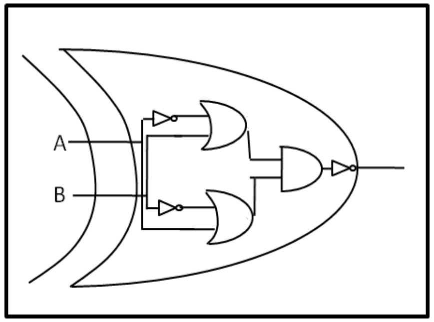


Verilog Question 2

- (Gate-level) 1-bit full adder
- **Step 1**: Create an XOR module



Design 1



Design 2

- **Step 2**: Implement the function of a one-bit full adder as follows:
  - Sum = $A \oplus B \oplus C\_in$
  - C_out = $A \cdot B + A \cdot C\_in + B \cdot C\_in$

Agenda

- Lab 1 Outline
- Lab 1 Basic Questions
- Lab 1 Advanced Questions
- Basic Concept of Verilog Testbench

Advanced Questions

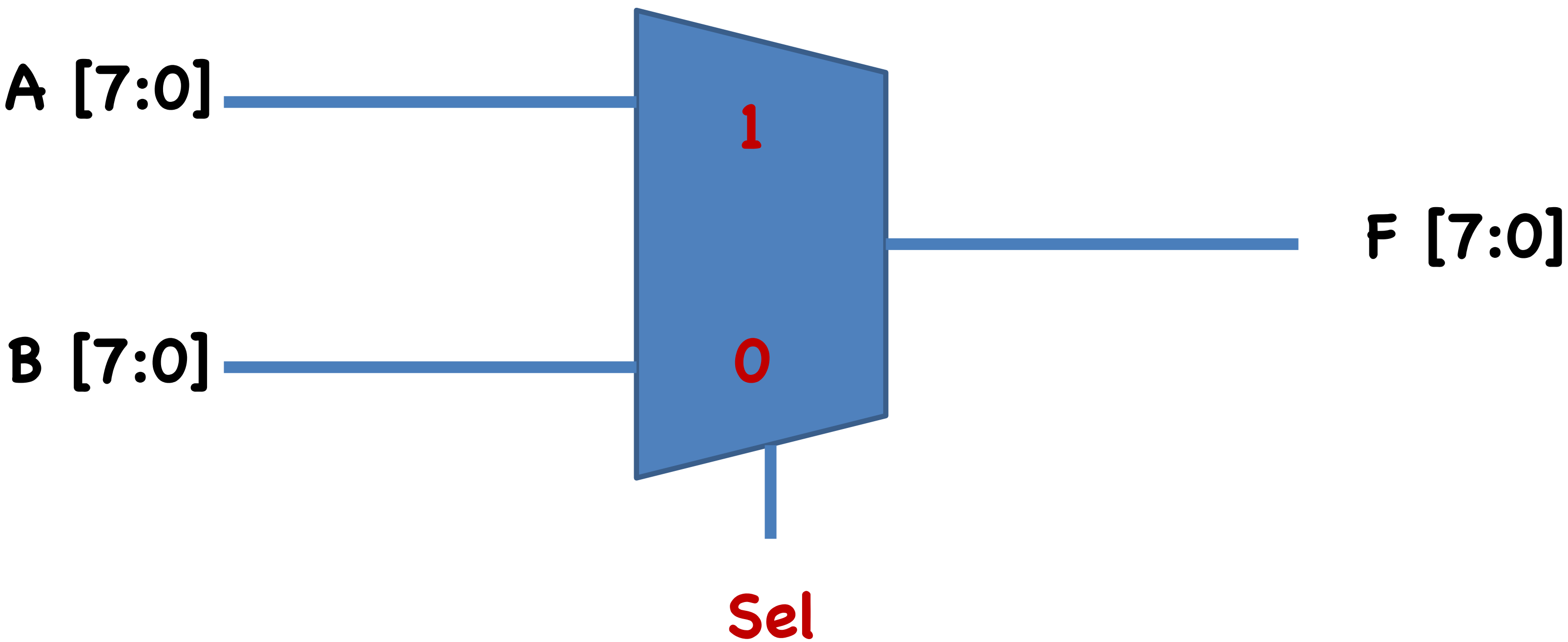
- Group assignment

- Verilog questions (due on 9/19/2019. 23:59:59.)
  - (Gate-level) 8-bit 2-to-1 MUX
  - (Gate-level) 4x16 decoder
  - (Gate-level) 3-bit comparator
  - (Gate-level) 4-bit ripple-carry adder (RCA)

- FPGA demonstration (due on 9/19/2019. In class.)
  - (Gate-level) 3-bit comparator

Verilog Question 1

- (Gate-level) 8-bit 2-to-1 MUX
- Instantiate 2-to-1 MUX modules from **Basic Question 1**



Verilog Question 2

- (Gate-level) 4x16 decoder

Input Din[3:0]	Output Dout[15:0]	Input Din[3:0]	Output Dout[15:0]
1111	0000_0000_0000_0001	0111	1000_0000_0000_0000
1110	0000_0000_0000_0010	0110	0100_0000_0000_0000
1101	0000_0000_0000_0100	0101	0010_0000_0000_0000
1100	0000_0000_0000_1000	0100	0001_0000_0000_0000
1011	0000_0000_0001_0000	0011	0000_1000_0000_0000
1010	0000_0000_0010_0000	0010	0000_0100_0000_0000
1001	0000_0000_0100_0000	0001	0000_0010_0000_0000
1000	0000_0000_1000_0000	0000	0000_0001_0000_0000

Verilog Question 3

- (Gate-level) 3-bit comparator
  - The 3-bits are unsigned numbers
  - No conditional operators, GATE LEVEL ONLY

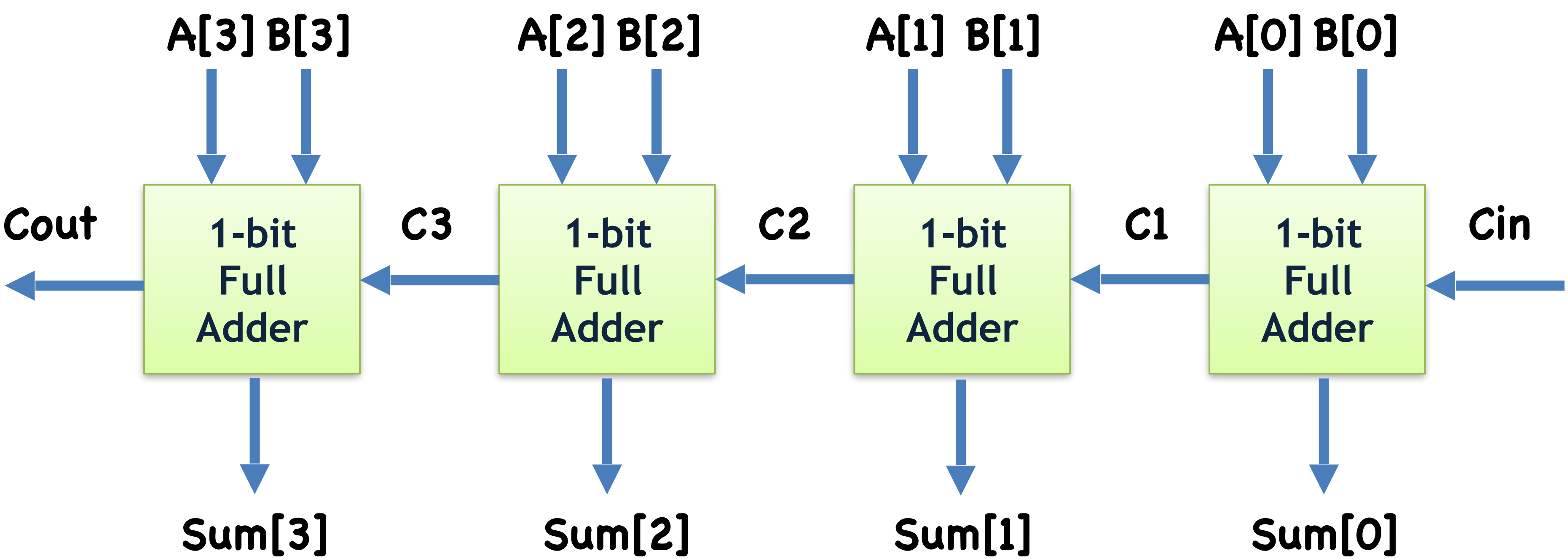
A_lt_B	Condition
1’b1	A[2:0] < B[2:0]
1’b0	Otherwise

A_gt_B	Condition
1’b1	A[2:0] > B[2:0]
1’b0	Otherwise

A_eq_B	Condition
1’b1	A[2:0] == B[2:0]
1’b0	Otherwise

Verilog Question 4

- (Gate-level) 4-bit ripple-carry adder (RCA)
- Instantiate the Full Adder module from Basic Question 2



Advanced Questions

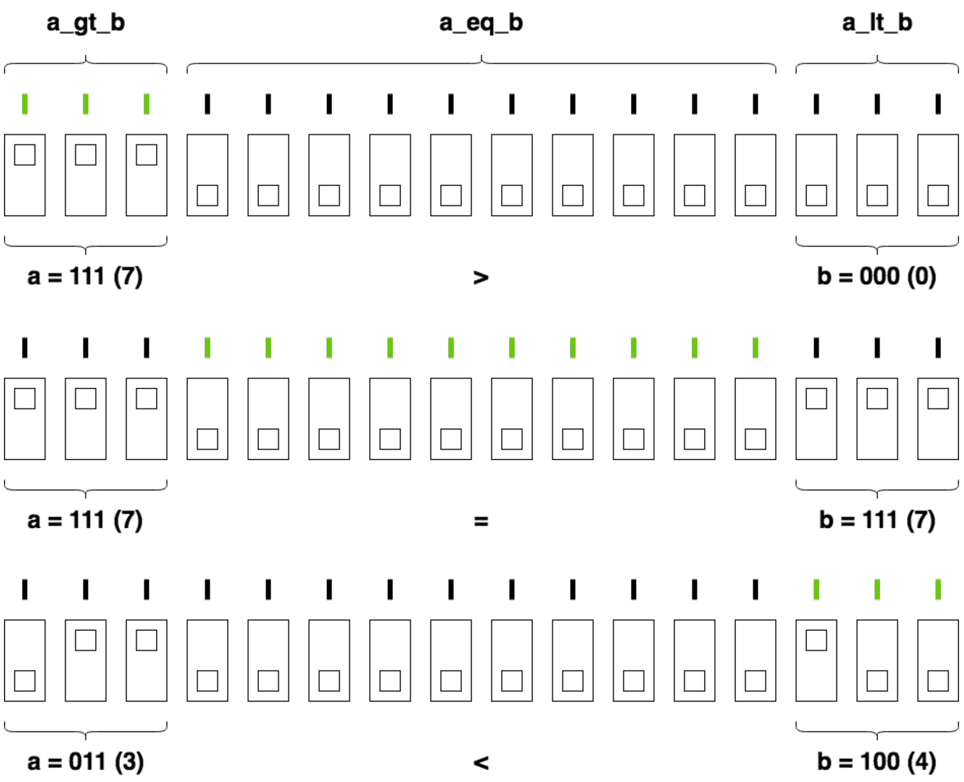
- Group assignment

- Verilog questions (due on 9/19/2019. 23:59:59.)
  - (Gate-level) 8-bit 2-to-1 MUX
  - (Gate-level) 4x16 decoder
  - (Gate-level) 3-bit comparator
  - (Gate-level) 4-bit ripple-carry adder (RCA)

- FPGA demonstration (due on 9/19/2019. In class)
  - (Gate-level) 3-bit comparator

FPGA Demonstration 1

- (Gate-level) 3-bit comparator
  - Please implement your gate-level 3-bit on your FPGA board
  - Please use SWITCHes as your inputs, and LEDs as your outputs
  - Please assign your inputs/outputs as:
    - A, B: The leftmost and rightmost three SWITCHes , respectively
    - A_lt_B, A_eq_B, A_gt_B: LEDs
    - **An example is** illustrated on the right hand side

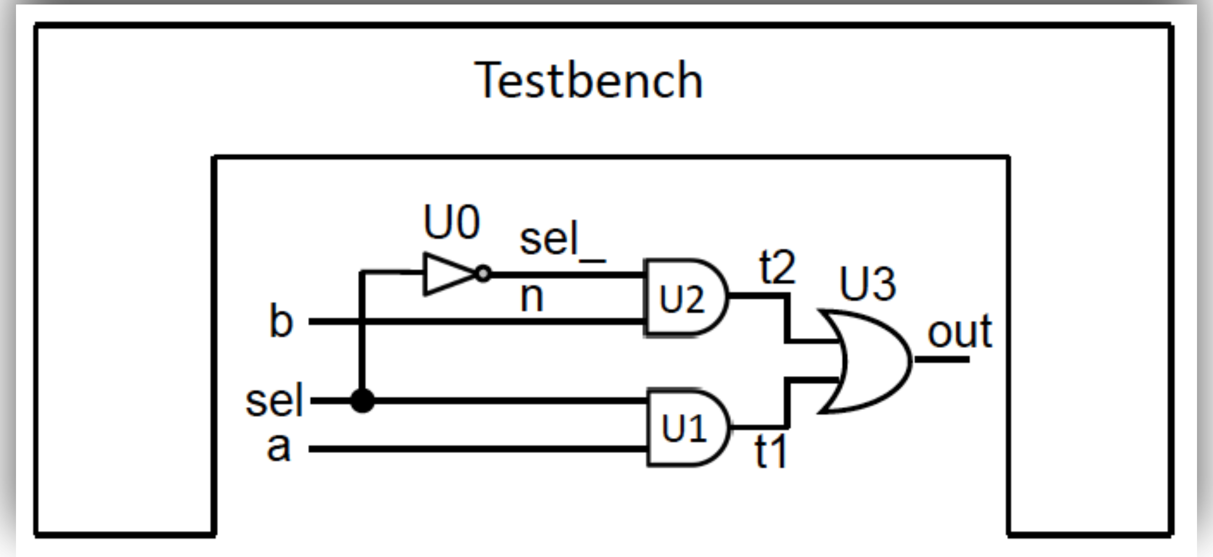


Agenda

- Lab 1 Outline
- Lab 1 Basic Questions
- Lab 1 Advanced Questions
- Basic Concept of Verilog Testbench

Verilog Simulation Framework

- Testbench verifies whether a module is correct or not
- Similar to the main function in C++
- Generate stimulus and check the outputs

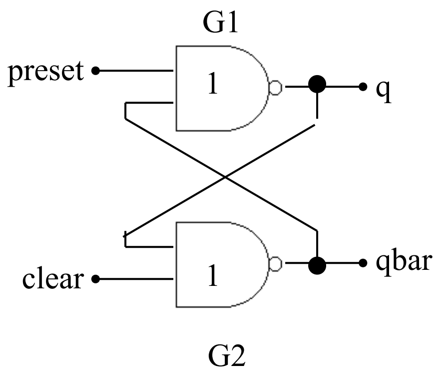


Verilog Testbench

Design

```
module   Nand_Latch_1 (q, qbar, preset, clear);
   output      q, qbar;
   input       preset, clear;

   nand #1     G1 (q, preset, qbar),
               G2 (qbar, clear, q);
endmodule
```



Testbench

```
`timescale 1ns / 1ps                          // Simulation Unit / Accuracy
module        test_Nand_Latch_1;              // Testbench module
 reg          preset, clear;                  // Inputs should be declared as reg
 wire         q, qbar;                        // Outputs should be declared as wire

 Nand_Latch_1 M1 (q, qbar, preset, clear);    // Instantiate YOUR DESIGN module

 always begin                                 // always condition: The description always happens
   #20        clear = !clear;                 // The value of clear inverts every 20 ns
 end

 initial                                      // Initial conditions
   begin
              preset = 1'b0;   clear = 1'b1;
   #10                                        // Units of "Simulation Units".  In this case, 10ns
              preset = 1'b1;
 end

endmodule
```

Thank you for your attention!

*Seattle night view taken at LA County Museum of Arts, Los Angeles, CA.
This picture is taken by Chun-Yi Lee himself, who is also a fan of photography